Docket No. 239415US2S

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

IN RE APPLICATION OF: Takanori TAMAI

GAU:

SERIAL NO: NEW APPLICATION

EXAMINER:

FILED: HEREWITH

FOR: INTEGRATED CIRCUIT DESIGN SYSTEM AND METHOD USING PREPROCESSOR WHICH CHANGES HARDWARE DESCRIPTION IN ACCORDANCE WITH CONFIGURATION

REQUEST FOR PRIORITY

COMMISSIONER FOR PATENTS
ALEXANDRIA, VIRGINIA 22313

SIR:

☐ Full benefit of the filing date of U.S. Application Serial Number , filed , is claimed pursuant to the provisions of **35 U.S.C. §120**.

☐ Full benefit of the filing date(s) of U.S. Provisional Application(s) is claimed pursuant to the provisions of **35 U.S.C. §119(e)**: **Application No.** **Date Filed**

■ Applicants claim any right to priority from any earlier filed applications to which they may be entitled pursuant to the provisions of **35 U.S.C. §119**, as noted below.

In the matter of the above-identified application for patent, notice is hereby given that the applicants claim as priority:

COUNTRY	APPLICATION NUMBER	MONTH/DAY/YEAR
Japan	2003-096687	March 31, 2003

Certified copies of the corresponding Convention Application(s)

■ are submitted herewith

☐ will be submitted prior to payment of the Final Fee

☐ were filed in prior application Serial No. filed

☐ were submitted to the International Bureau in PCT Application Number
Receipt of the certified copies by the International Bureau in a timely manner under PCT Rule 17.1(a) has been acknowledged as evidenced by the attached PCT/IB/304.

☐ (A) Application Serial No.(s) were filed in prior application Serial No. filed ; and

☐ (B) Application Serial No.(s)

☐ are submitted herewith

☐ will be submitted prior to payment of the Final Fee

Respectfully Submitted,

OBLON, SPIVAK, McCLELLAND,
MAIER & NEUSTADT, P.C.

Marvin J. Spivak
Registration No. 24,913

C. Irvin McClelland
Registration Number 21,124

22850

Tel. (703) 413-3000
Fax. (703) 413-2220
(OSMMN 05/03)

02S0643-1

日　本　国　特　許　庁
JAPAN　PATENT　OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出願年月日
Date of Application:　２００３年　３月３１日

出願番号
Application Number:　特願２００３－０９６６８７

[ST.10/C]:　[ＪＰ２００３－０９６６８７]

出願人
Applicant(s):　株式会社東芝

２００３年　４月１８日

特許庁長官
Commissioner,
Japan Patent Office　太田信一郎

出証番号　出証特２００３－３０２８１６１

【書類名】　特許願

【整理番号】　A000202777

【提出日】　平成15年 3月31日

【あて先】　特許庁長官　殿

【国際特許分類】　G06F 17/50

【発明の名称】　プリプロセッサ、集積回路の設計システム及び集積回路の設計方法

【請求項の数】　20

【発明者】

【住所又は居所】　神奈川県川崎市幸区小向東芝町１番地　株式会社東芝マイクロエレクトロニクスセンター内

【氏名】　玉井　孝典

【特許出願人】

【識別番号】　000003078

【氏名又は名称】　株式会社　東芝

【代理人】

【識別番号】　100058479

【弁理士】

【氏名又は名称】　鈴江　武彦

【電話番号】　03-3502-3181

【選任した代理人】

【識別番号】　100091351

【弁理士】

【氏名又は名称】　河野　哲

【選任した代理人】

【識別番号】　100088683

【弁理士】

【氏名又は名称】　中村　誠

【選任した代理人】

【識別番号】　100108855

【弁理士】

【氏名又は名称】　蔵田　昌俊

【選任した代理人】

【識別番号】　100084618

【弁理士】

【氏名又は名称】　村松　貞男

【選任した代理人】

【識別番号】　100092196

【弁理士】

【氏名又は名称】　橋本　良郎

【手数料の表示】

【予納台帳番号】　011567

【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　プリプロセッサ、集積回路の設計システム及び集積回路の設計方法

【特許請求の範囲】

【請求項１】　プリプロセッサ制御ファイルに基づいて、第１のハードウェア記述言語と第２のハードウェア記述言語が混在する第１の回路記述ファイルの処理を行い、前記第１の回路記述ファイルにおける第１のハードウェア記述言語で記述された少なくとも一部を第２のハードウェア記述言語に変換して第２の回路記述ファイルを生成し、出力することを特徴とするプリプロセッサ。

【請求項２】　前記プリプロセッサ制御ファイルと前記第１の回路記述ファイルとに基づいて、ゲーテッドクロック回路用の論理合成制御スクリプトファイルを更に生成して出力することを特徴とする請求項１に記載のプリプロセッサ。

【請求項３】　前記第２のハードウェア記述言語は、Ｖｅｒｉｌｏｇ－ＨＤＬまたはＶＨＤＬを含むことを特徴とする請求項１または２に記載のプリプロセッサ。

【請求項４】　前記第１の回路記述ファイルの処理は、前記第１の回路記述ファイルから前記第１のハードウェア記述言語を抽出して前記第２のハードウェア記述言語に変換し、前記第１の回路記述ファイル中の前記第２のハードウェア記述言語で記述された部分は変換せずに出力するものであることを特徴とする請求項１乃至３いずれか１つの項に記載のプリプロセッサ。

【請求項５】　前記第１の回路記述ファイルは、第１のハードウェア記述言語で記述されたフリップフロップに関する記述を含み、前記フリップフロップの記述をゲーテッドクロック化した回路に対応する情報を第２のハードウェア記述言語に変換して生成し、出力することを特徴とする請求項１乃至４いずれか１つの項に記載のプリプロセッサ。

【請求項６】　前記第１のハードウェア記述言語で記述されたフリップフロップに関する記述は、フリップフロップのリセット方式が定まっていない状態で記述され、前記第１の回路記述ファイルにおける第１のハードウェア記述言語で記述された少なくとも一部を第２のハードウェア記述言語に変換する際に、前記

フリップフロップが同期リセット方式か非同期リセット方式かを指定することを特徴とする請求項５に記載のプリプロセッサ。

【請求項７】　前記フリップフロップに関する記述は、１つのフリップフロップ記述に１つのクラスタ番号が付与されていることを特徴とする請求項５または６に記載のプリプロセッサ。

【請求項８】　前記第２の回路記述ファイルは、複数のクラスタ番号で示されたフリップフロップの記述が、１つのゲーティングされたクロック信号で駆動されるフリップフロップの記述に変換されたものであることを特徴とする請求項５乃至７いずれか１つの項に記載のプリプロセッサ。

【請求項９】　プリプロセッサ制御ファイルに基づいて、第１のハードウェア記述言語で記述されたフリップフロップの記述を含む第１の回路記述ファイルの処理を行い、少なくとも前記フリップフロップの記述を第２のハードウェア記述言語に変換して第２の回路記述ファイルを生成し、且つゲーテッドクロック回路用の論理合成制御スクリプトファイルを生成するプリプロセッサと、

前記プリプロセッサで生成された前記第２の回路記述ファイルと前記論理合成制御スクリプトファイルとを論理合成して、回路を構成する基本単位であるセルを使用した回路記述ファイルに変換することによりネットリストを生成し、前記ネットリストに基づいて集積回路におけるセルと配線の配置を決定する論理合成ツールと

を具備することを特徴とする集積回路の設計システム。

【請求項１０】　前記ネットリストは、前記第２の回路記述ファイルと前記論理合成制御スクリプトファイルに加えて、更にゲーテッドクロック以外の制御を行うための論理合成制御スクリプトファイルを論理合成して生成されることを特徴とする請求項９に記載の集積回路の設計システム。

【請求項１１】　前記論理合成ツールによる論理合成によって、フリップフロップのゲーテッドクロック化のための情報を生成することを特徴とする請求項９または１０に記載の集積回路の設計システム。

【請求項１２】　前記第１のハードウェア記述言語で記述されたフリップフロップに関する記述は、フリップフロップのリセット方式が定まっていない状態

で記述され、前記第１の回路記述ファイルにおける第１のハードウェア記述言語で記述された少なくとも一部を第２のハードウェア記述言語に変換する際に、前記フリップフロップが同期リセット方式か非同期リセット方式かが指定されることを特徴とする請求項９乃至１１いずれか１つの項に記載の集積回路の設計システム。

【請求項１３】　前記フリップフロップに関する記述は、１つのフリップフロップ記述に１つのクラスタ番号が付与されることを特徴とする請求項９乃至１２いずれか１つの項に記載の集積回路の設計システム。

【請求項１４】　前記第２の回路記述ファイルは、複数のクラスタ番号で示されるフリップフロップの記述が、１つのゲーティングされたクロック信号で駆動されるフリップフロップの記述に変換されたものであることを特徴とする請求項９乃至１２いずれか１つの項に記載の集積回路の設計システム。

【請求項１５】　第１のハードウェア記述言語と第２のハードウェア記述言語が混在する回路記述ファイルと、プリプロセッサの動作を制御するためのプリプロセッサ制御ファイルとをプリプロセッサに入力し、前記回路記述ファイルにおける第１のハードウェア記述言語で記述された少なくとも一部を第２のハードウェア記述言語に変換するステップと、

前記プリプロセッサから出力される回路記述ファイルとゲーテッドクロック回路用の論理合成制御スクリプトファイル、及びゲーテッドクロック回路以外の論理合成制御スクリプトファイルを論理合成ツールで論理合成し、回路を構成する基本単位であるセルを使用した回路記述ファイルに変換してネットリストを生成するステップと、

前記ネットリストに基づいて前記セルと配線の配置を決定し、チップの回路設計を行うステップと

を具備することを特徴とする集積回路の設計方法。

【請求項１６】　前記回路記述ファイルは、第１のハードウェア記述言語で記述されたフリップフロップの記述を含み、前記プリプロセッサによって同期／非同期指定とゲーテッドクロックのクラスタ結合が変更されることを特徴とする請求項１５に記載の集積回路の設計方法。

【請求項１７】　前記プリプロセッサ制御ファイルは、リセット信号の名前、クロック信号の名前、フリップフロップのリセット方式、ゲーテッドクロック化するか否か、ゲーテッドクロック化する場合に、そのクロックゲーティングの記述をプリプロセッサが生成するか否か、論理合成プログラムの自動ゲーテッドクロック化機能が自動認識できる記述形式で出力するか否か、及びどのクラスタを１つと見なすかに関する情報の少なくともいずれか１つを含むことを特徴とする請求項１６に記載の集積回路の設計方法。

【請求項１８】　前記回路記述ファイルにおける第１のハードウェア記述言語で記述された少なくとも一部を第２のハードウェア記述言語に変換するステップは、

前記プリプロセッサ制御ファイルを読み込んで解釈し、この解釈した情報を前記プリプロセッサに格納するステップと、

前記プリプロセッサに格納した情報が拡張記述に関するか否か判定し、拡張記述であったときに拡張記述の解析を行うステップと、

拡張記述でないと判定されたときにタグの判定を行うステップと、

所定のタグであると判定されたときに、クラスタ結合指定の判定を行うステップと、

クラスタ結合指定がないときに、フリップフロップを同期リセットにするか非同期リセットにするかを指定し、同期リセットのフリップフロップに対応する回路記述または非同期リセットのフリップフロップに対応する回路記述を生成するステップと、

クラスタ結合指定があるときに、クラスタ結合に必要な情報を生成するステップと

を備えることを特徴とする請求項１５乃至１７いずれか１つの項に記載の集積回路の設計方法。

【請求項１９】　前記拡張記述の解析を行うステップは、

フリップフロップに関する拡張記述か否かを判定するステップと、

前記判定ステップでフリップフロップに関する拡張記述であると判定されたときに、構造体フリップフロップの各変数へ情報を格納するステップと、

前記構造体フリップフロップの実体へのポインタに対してタグを付加し、一時バッファに格納するステップと

を備えることを特徴とする請求項１８に記載の集積回路の設計方法。

【請求項２０】　前記タグの判定を行うステップは、

前記一時バッファからタグ及び格納情報を取得するステップと、

前記タグが所定のタグか否か判定し、所定のタグではないときに前記格納情報をそのまま出力ファイルへ出力するステップと

を備えることを特徴とする請求項１８に記載の集積回路の設計方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

この発明は、ハードウェア記述をコンフィギュレーションに従って変更する作業を自動化する場合などにおいて使用されるプリプロセッサ、このプリプロセッサを用いた集積回路の設計システム、及びこのシステムによる集積回路の設計方法に関する。特にＩＰ，ソフトＩＰをユーザの指定に従って自動的に生成するシステムなどに好適なもので、フリップフロップの同期／非同期指定やゲーテッドクロックのクラスタ結合を変更可能な回路記述ファイルを生成するためのものである。

【０００２】

【従来の技術】

従来、集積回路の設計システムは、例えば図１３に示すように構成されている。このような集積回路を設計するためのコンピュータシステムとその方法については、例えば特許文献１に記載されている。

【０００３】

ユーザによって既存の言語（Ｖｅｒｉｌｏｇ－ＨＤＬ:Verilog-Hardware Description LanguageまたはＶＨＤＬ:Very High Speed Integrated Circuits Hardware Description Languageなど）で記述された回路記述ファイル１１と、論理合成制御スクリプトファイル（論理合成プログラムを制御するためのスクリプトファイル）１２がプロセッサ（コンピュータ）１３の論理合成プログラム１４によ

って論理合成される。論理合成プログラム１４では、上記回路記述ファイル１１を、回路を構成する基本単位である「セル」を使用した回路記述ファイルに変換する処理が行われ、ネットリスト（Netlist）１５が生成される。このネットリスト１５が配置配線プログラム（セル及び配線を配置するプログラム）１６により処理されて、セルと配線の配置が決定され、チップ（半導体集積回路）１７の回路設計が行われる。

【０００４】

ところで、上記のような従来の集積回路の設計システムでは、Ｖｅｒｉｌｏｇ－ＨＤＬやＶＨＤＬなどの既存の言語を使用して同期リセット回路や非同期リセット回路を記述する場合には、同期リセット用と非同期リセット用の２つの記述を人手で作成しなければならない。また、低消費電力化のためにフリップフロップのゲーテッドクロック化を行う場合、何個のフリップフロップを１つのクラスタとするのが最適であるか、あるいはどのフリップフロップとどのフリップフロップがチップ中で近くに配置されているからクラスタ結合をすべきか、などの条件が種々の要因で異なるため最適なクラスタの構成方法が異なってくる。上記要因としては、例えば下記（１）～（３）がある。

【０００５】

（１）半導体集積回路（ＩＣやＬＳＩ）の物理的なテクノロジの相違。

【０００６】

（２）回路の動作パターン、すなわちユーザアプリケーションがその回路をどのように使用するかの違い。

【０００７】

（３）チップ中での各回路の配置の仕方。

【０００８】

しかし、従来は人手により、同期リセット用と非同期リセット用の記述を作成したり、ゲーテッドクロック化のための回路を試行錯誤で挿入していたので、最適値に収束させるのが難しい。しかも、ゲーテッドクロック化のための回路を挿入する場合には周辺の回路を修正する必要があり、フリップフロップの記述周辺の修正だけとはいえ、回路規模が大きいと数千ヶ所になることもある。また、人

手による修正では、バグの混入を避けることができず、修正のたびに機能検証作業が必要となる。ＬＳＩの回路規模や複雑度は大変な速度で上昇しており、これに伴って検証作業にかかる時間も膨大になっている。このため、開発期間の長期化やコストの上昇の１つの要因になっている。

【０００９】

なお、上記ゲーテッドクロック化のための回路を自動的に挿入するツールは既に存在しているが（例えば非特許文献１参照）、指定したクラスタを結合するなどの細かな要求に応えることはできず、必ずしも満足できるものではない。

【００１０】

【特許文献１】

U.S.Patent No.5,987,239. Graham Kirsch "COMPUTER SYSTEM AND METHOD FOR BUILDING A HARDWARE DESCRIPTION LANGUAGE REPRESENTATION OF CONTROL LOGIC FOR A COMPLEX DIGITAL SYSTEM" Nov.16,1996.

【００１１】

【非特許文献１】

0-8186-4350-1/93 1993 IEEE pp.466-471 "A Workbench for Generation of Component Models" Marcus Bluml et al.

【００１２】

【発明が解決しようとする課題】

上述したように、従来のプリプロセッサ、集積回路の設計システム及び集積回路の設計方法では、人手による作業が必要なため、時間がかかりバグの混入も不可避であり、開発期間の長期化やコストの上昇の要因となる、という問題があった。

【００１３】

また、ゲーテッドクロック化のための回路を自動的に挿入するツールは存在するが、細かな要求に応えることができない、という問題があった。

【００１４】

この発明は上記のような事情に鑑みてなされたもので、その目的とするところは、フリップフロップの同期／非同期指定及びゲーテッドクロックのクラスタ結

合を変更可能な回路記述ファイルを生成できるプリプロセッサを提供することにある。

【００１５】

また、開発期間を短縮し、コストを低減できる集積回路の設計システムとこのシステムによる集積回路の設計方法を提供することにある。

【００１６】

【課題を解決するための手段】

この発明の一態様によると、プリプロセッサ制御ファイルに基づいて、第１のハードウェア記述言語と第２のハードウェア記述言語が混在する第１の回路記述ファイルの処理を行い、前記第１の回路記述ファイルにおける第１のハードウェア記述言語で記述された少なくとも一部を第２のハードウェア記述言語に変換して第２の回路記述ファイルを生成し、出力するプリプロセッサが提供される。

【００１７】

上記のような構成のプリプロセッサを用いれば、人手を介在することなく、フリップフロップの同期／非同期指定やゲーテッドクロックのクラスタ結合を変更可能な回路記述ファイル（第２の回路記述ファイル）を生成できる。よって、人手による作業時間を短縮するとともに、バグの混入を避けることができ、検証作業にかける時間も不要になる。

【００１８】

また、この発明の一態様によると、プリプロセッサ制御ファイルに基づいて、第１のハードウェア記述言語で記述されたフリップフロップの記述を含む第１の回路記述ファイルの処理を行い、少なくとも前記フリップフロップの記述を第２のハードウェア記述言語に変換して第２の回路記述ファイルを生成し、且つゲーテッドクロック回路用の論理合成制御スクリプトファイルを生成するプリプロセッサと、前記プリプロセッサで生成された前記第２の回路記述ファイルと前記論理合成制御スクリプトファイルとを論理合成して、回路を構成する基本単位であるセルを使用した回路記述ファイルに変換することによりネットリストを生成し、前記ネットリストに基づいて集積回路におけるセルと配線の配置を決定する論理合成ツールとを具備する集積回路の設計システムが提供される。

【００１９】

上記のような構成のシステムによれば、プリプロセッサでフリップフロップの同期／非同期指定やゲーテッドクロックのクラスタ結合を変更可能な回路記述ファイル（第２の回路記述ファイル）を生成するので、従来は人手による作業で行っていた部分を自動化して集積回路を設計できる。これによって、作業時間を短縮し、且つ人手の介在によるバグの混入を避けることができ、検証作業にかける時間も不要になる。よって、開発期間を短縮し、コストを低減できる。

【００２０】

更に、この発明の一態様によると、第１のハードウェア記述言語と第２のハードウェア記述言語が混在する回路記述ファイルと、プリプロセッサの動作を制御するためのプリプロセッサ制御ファイルとをプリプロセッサに入力し、前記回路記述ファイルにおける第１のハードウェア記述言語で記述された少なくとも一部を第２のハードウェア記述言語に変換するステップと、前記プリプロセッサから出力される回路記述ファイルとゲーテッドクロック回路用の論理合成制御スクリプトファイル、及びゲーテッドクロック回路以外の論理合成制御スクリプトファイルを論理合成ツールで論理合成し、回路を構成する基本単位であるセルを使用した回路記述ファイルに変換してネットリストを生成するステップと、前記ネットリストに基づいて前記セルと配線の配置を決定し、チップの回路設計を行うステップとを具備する集積回路の設計方法が提供される。

【００２１】

上記のような方法によれば、従来は人手による作業で行っていた部分を自動化して集積回路を設計できる。これによって、作業時間を短縮し、且つ人手の介在によるバグの混入を避けることができ、検証作業にかける時間も不要になる。この結果、開発期間を短縮し、コストを低減できる。

【００２２】

【発明の実施の形態】

以下、この発明の実施の形態について図面を参照して説明する。

図１は、この発明の実施の形態に係るプリプロセッサ及び集積回路の設計システムについて説明するためのもので、システム全体の概略構成を示している。ユ

ーザによって記述された回路記述ファイル２１と、同じくユーザによって記述されたプリプロセッサ制御ファイル（プリプロセッサの動作を制御するための記述ファイル）２２がプリプロセッサ２３に入力され、プリプロセッサ制御ファイル２２の制御データに基づいて回路記述ファイル２１に対する処理が行われる。上記回路記述ファイル２１には、異なるハードウェア記述言語（ここでは記述言語Ａと記述言語Ｂと称する）が混在している。また、上記回路記述ファイル２１には、下記（ａ）～（ｆ）の情報が記述されている。

【００２３】

（ａ）リセット信号の名前

（ｂ）クロック信号の名前

（ｃ）フリップフロップのリセット方式［同期／非同期］

（ｄ）ゲーテッドクロック化するか否か［ＹＥＳ／ＮＯ］

（ｅ）ゲーテッドクロック化する場合に、そのクロックゲーティングの記述をプリプロセッサが生成するか否か

論理合成プログラムの自動ゲーテッドクロック化機能が自動認識できる記述形式で出力するか否か［自動／プリプロセッサが生成］

（ｆ）どのクラスタを１つと見なすか（結合）に関する情報（複数可）

上記プリプロセッサ２３により、上記回路記述ファイル２１中の記述言語Ａが既存の記述言語Ｂ（Ｖｅｒｉｌｏｇ－ＨＤＬまたはＶＨＤＬなど）に変換され、回路記述ファイル２４が作成される。また、論理合成プログラム２５でゲーテッドクロック化を行う際に使用する論理合成制御スクリプトファイル（ゲーテッドクロック回路用）２６が作成される。

【００２４】

論理合成ツール（コンピュータ）２７には、上記回路記述ファイル２４及び上記論理合成制御スクリプトファイル２６に加えて、予め用意されているゲーテッドクロックに関するもの以外の論理合成スクリプトファイル２８が入力される。論理合成プログラム２５では、上記回路記述ファイル２４を、回路を構成する基本単位であるセルを使用した回路記述ファイルに変換する処理が行われる。この段階では、セル及び配線をどのように配置するかは決まっていない。そして、こ

の論理合成プログラム２５によってネットリスト（論理合成プログラムから出力されたセルを使用した回路記述）２９が生成される。

【００２５】

上記ネットリスト２９は、配置配線プログラム（セル及び配線を配置するプログラム）３０により処理されてセルと配線の配置が決定される。そして、この配置配線プログラム３０の処理結果に基づいて、チップ（半導体集積回路）３１の回路設計が行われる。

【００２６】

図２乃至図７はそれぞれ、上記図１に示した集積回路の設計システムにおけるプリプロセッサ２３の動作を示すフローチャートである。図２は図１に示したシステムにおけるプリプロセッサの概略的な動作を示すフローチャート、図３は第１ステージの処理サブルーチンを示すフローチャート、図４は拡張記述解析サブルーチンを示すフローチャート、図５は第２ステージの処理サブルーチンを示すフローチャート、図６はタグＢの処理サブルーチンを示すフローチャート、図７はクラスタ結合サブルーチンを示すフローチャートである。

【００２７】

図２に示すように、まず、プリプロセッサ２３のプログラム本体の動作が開始されると、プリプロセッサ制御ファイルの読み込みとその解釈が行われ、その情報が配列ＣＴＲＬへ格納される（ＳＴＥＰ１）。

【００２８】

次に、図３に示す第１ステージの処理サブルーチンに入る（ＳＴＥＰ２）。第１ステージの処理では、ユーザによる回路記述ファイル２１を開き（ＳＴＥＰ３）、ファイルが終わりか否かを判定する（ＳＴＥＰ４）。ファイルが終わりでないと判定されると、拡張記述に関する記述か否かが判定される（ＳＴＥＰ５）。ここで拡張記述であると判定されると拡張記述解析サブルーチンへジャンプし（ＳＴＥＰ６）、拡張記述でないと判定されると、何も変換せずに一時バッファｔｍｐにタグ「Ａ」を付加して格納し（ＳＴＥＰ７）、ＳＴＥＰ４の判定動作に戻る。そして、ファイルが終わりと判定されるまで上述したＳＴＥＰ４乃至ＳＴＥＰ７の動作を繰り返す。ＳＴＥＰ４でファイルが終わりと判定されると、サブル

ーチンの呼び出し元（ＳＴＥＰ２）へ戻る。

【００２９】

拡張記述サブルーチンでは、図４に示すように、フリップフロップに関する拡張記述か否かが判定される（ＳＴＥＰ８）。そして、フリップフロップに関する拡張記述であると判定されると、拡張回路記述の文法解析により情報を構造体ＦＦの各変数へ格納し（ＳＴＥＰ９）、フリップフロップに関する拡張記述でないと判定されると、エラー処理を行ってプログラムを終了する（ＳＴＥＰ１０）。上記構造体ＦＦの各変数への情報の格納は、変数名Ｎａｍｅにフリップフロップの名前、ＢｉｔＮｕｍにフリップフロップのビット数、ＣｌｕｓｔｅｒＮｏにクラスタ番号、ＣｔｒｌＮａｍｅＰｔｒ＊にフリップフロップの制御信号群の名前へのポインタ、ＶａｌＰｔｒ＊にフリップフロップの制御信号群がそれぞれ１になった場合に代入すべき値または信号群へのポインタの情報をそれぞれ格納する。

【００３０】

次に、構造体ＦＦの実体へのポインタに、タグ「Ｂ」を付加して一時バッファｔｍｐに格納する（ＳＴＥＰ１１）。

【００３１】

その後、サブルーチンの呼び出し元（ＳＴＥＰ５）へ戻る。

【００３２】

上述した第１ステージの処理サブルーチンが終了すると、第２ステージの処理サブルーチンへジャンプする（ＳＴＥＰ１２）。第２ステージの処理サブルーチンでは、まず図５に示すように、一時バッファｔｍｐの終わりか否かが判定され（ＳＴＥＰ１３）、終わりでないと判定されると一時バッファｔｍｐからタグ及び格納情報を取得する（ＳＴＥＰ１４）。そして、タグが「Ｂ」か否かが判定される（ＳＴＥＰ１５）。タグが「Ｂ」でない場合には、格納情報をそのまま出力ファイルＯＵＴに出力（ＳＴＥＰ１６）してＳＴＥＰ１３に戻って同様な処理を繰り返す。そして、上記ＳＴＥＰ１３で一時バッファｔｍｐの終わりと判定されると、サブルーチンの呼び出し元（ＳＴＥＰ１２）へ戻る。一方、タグが「Ｂ」であった場合には、タグＢの処理サブルーチンへジャンプする（ＳＴＥＰ１７）

。

【００３３】

タグＢの処理サブルーチンは、図６に示すように、配列ＣＴＲＬにゲーテッドクロック機能を有効にする指定が存在し、タグＢとともに格納された構造体ＦＦのデータにおける変数ＣｌｕｓｔｅｒＮｏ（＝ＣｕｒｒｅｎｔＮｏ）と別のＣｌｕｓｔｅｒＮｏ（＝ＡｐｐｅｎｄＮｏ）とのクラスタ結合指定が存在するか否かが判定される（ＳＴＥＰ１８）。クラスタ結合指定が存在する場合には、クラスタ結合サブルーチンへジャンプし（ＳＴＥＰ１９）、存在しない場合には配列ＣＴＲＬにゲーテッドクロック用セルを直接配置する指定が存在するか否かが判定される（ＳＴＥＰ２０）。指定が存在する場合には、フリップフロップ用のクロック信号として、配列ＣＴＲＬに格納されているクロック信号をゲーテッドクロック用に制御する回路を通過した後の信号名を使用するように変更する。また、制御する回路用のＶｅｒｉｌｏｇ記述を出力ファイルＯＵＴに出力する（ＳＴＥＰ２１）。一方、指定が存在しない場合には、フリップフロップ用のクロック信号として配列ＣＴＲＬに格納されているクロック信号を使用することにする（ＳＴＥＰ２２）。

【００３４】

上記ＳＴＥＰ２１及びＳＴＥＰ２２による処理結果に対して、配列ＣＴＲＬにフリップフロップを非同期リセットに設定する項目が存在するか否かが判定される（ＳＴＥＰ２３）。存在する場合には、構造体ＦＦの変数ＣｔｒｌＮａｍｅＰｔｒで示される配列に配列ＣＴＲＬに格納された「リセット信号」と同じものが存在した場合には、その信号を非同期リセット信号と見なして構造体ＦＦの内容を、非同期リセットを有するフリップフロップとなるようなＶｅｒｉｌｏｇ記述を生成し、出力ファイルＯＵＴへ出力する（ＳＴＥＰ２４）。これに対し、存在しない場合には、構造体ＦＦの変数ＣｔｒｌＮａｍｅＰｔｒで示される配列に配列ＣＴＲＬに格納された「リセット信号」と同じものが存在した場合には、その信号を同期リセット信号と見なして構造体ＦＦの内容を、同期リセットを有するフリップフロップとなるようなＶｅｒｉｌｏｇ記述を生成し、出力ファイルＯＵＴへ出力する（ＳＴＥＰ２５）。その後、サブルーチンの呼び出し元（ＳＴＥＰ

１５）に戻る。

【００３５】

上記ＳＴＥＰ１９において、クラスタ結合サブルーチンへジャンプした場合には、図７に示すように、結合対象のＡｐｐｅｎｄＮｏが存在するか否かが判定され（ＳＴＥＰ２６）、存在する場合には、ＣｕｒｒｅｎｔＮｏの構造体ＦＦとＡｐｐｅｎｄＮｏの構造体を結合して、１つのゲーテッドクラスタを形成するようにＣｕｒｒｅｎｔＮｏの構造体ＦＦの構造を変更する（ＳＴＥＰ２７）。次に、結合に必要な回路のうち、構造体ＦＦだけで表現できないＶｅｒｉｌｏｇ記述に関する記述を生成して出力ファイルＯＵＴに出力する（ＳＴＥＰ２８）。その後、Ｖｅｒｉｌｏｇ記述に関する情報を一時バッファｔｍｐから削除する（ＳＴＥＰ２９）。そして、ＳＴＥＰ２６に戻って結合対象のＡｐｐｅｎｄＮｏが存在しなくなるまでＳＴＥＰ２７乃至ＳＴＥＰ２９の処理を繰り返し、存在しなくなるとゲーテッドクロック化の対象となったフリップフロップのレジスタ名などの情報を論理合成制御ファイルの形式で出力する（ＳＴＥＰ３０）。次に、サブルーチンの呼び出し元（ＳＴＥＰ１８）に戻る。

【００３６】

図８は、上記図２乃至図７に示したプリプロセッサによるフリップフロップのリセットの同期／非同期指定を変更する場合の入力ファイルと出力ファイルを示している。ここでは、フリップフロップのリセット方式として同期リセット／非同期リセットを指定した場合の、それぞれの変換プログラムＡ（Ｐｒｏｇｒａｍ Ａ）の出力結果を示す。

【００３７】

図８において、“＜／”と“／＞”はマーカー（ｍａｒｋｅｒ）であり、“＜／”と“／＞”で囲まれた部分の記述が変換対象となる。ｃｌｋ　ｃｌｏｃｋは変換後のハードウェア記述におけるクロック信号の名前（ここではｃｌｏｃｋ）を指定するものである。ｄｅｌａｙは変換後のハードウェア記述におけるシミュレーションのディレイ値（ここでは２）を指定するものである。

【００３８】

また、ｐｒｏｇｒａｍＡは、プリプロセッサ２３の変換プログラムＡを意味し

ており、ｏｐｔｉｏｎ－ＳＹＮＣは、上記ｐｒｏｇｒａｍＡに対して同期リセット方式でフリップフロップを初期化するハードウェア記述を出力することを指定し、ｏｐｔｉｏｎ－ＡＳＹＮＣは、上記ｐｒｏｇｒａｍＡに対して非同期リセット方式でフリップフロップを初期化するハードウェア記述を出力することを指定する。

【００３９】

上記のようなプリプロセッサへの入力ファイルに対して、フリップフロップのリセットの同期／非同期指定に応じた変換プログラムＡによる処理結果が出力される。

【００４０】

このように、プリプロセッサ２３により、フリップフロップの同期／非同期指定を自由に変更可能な回路記述ファイルを生成できる。また、フリップフロップのリセットに関して「同期リセット」とした記述と「非同期リセット」とした記述の２つは、変換プログラムＡを２回実行すれば得られる。これにより人手で作成した場合に比べて、短期間で「同期リセット」とした記述と「非同期リセット」の２つを得ることが可能である。

【００４１】

図９は、ゲーテッドクロック機能をグルーピング指定なしで行った場合（クラスタ結合機能を使わない場合）のプリプロセッサ（変換プログラムＡ）２３の入力ファイルと出力ファイルの例である。また、図１０は、上記図９に示した出力ファイルから形成された回路例を示している。ここで、ａ００，ａ０１はクロック信号を停止（ゲーティング）するための回路、ｂ００，ｂ０２はラッチ、ｂ０１，ｂ０３は論理積回路、ｃ００，ｃ０１，ｃ０２はフリップフロップ、ｄ００，ｄ０１はクラスタを表している。

【００４２】

プリプロセッサ２３にゲーテッドクロック機能を指示する信号ｅｎａｂｌｅｇａｔｅ－ｃｌｏｃｋが入力されると、入力ファイルは変換プログラムＡにより処理されて出力ファイルが生成される。

【００４３】

図１１は、ゲーテッドクロック機能をグルーピング指定して行った場合（クラスタ結合機能を使った場合）のプリプロセッサ（変換プログラムＡ）の入力ファイルと出力ファイルの例である。また、図１２は、上記図１１に示した出力ファイルから形成された回路例を示している。プリプロセッサ２３には、ゲーテッドクロック機能を指示する信号ｅｎａｂｌｅ　ｇａｔｅ－ｃｌｏｃｋとクラスタ結合機能を指示する信号ｇｒｏｕｐｉｎｇ｛ＦＦ００，ＦＦ０１｝が入力される。クラスタ結合機能は変換プログラムＡに対してｇｒｏｕｐｉｎｇ｛ＦＦ００，ＦＦ０１｝などを指定することで機能する。ここで、ａ００はクロック信号を停止（ゲーティング）するための回路、ｂ００はラッチ、ｂ０１は論理積回路、ｃ００，ｃ０１，ｃ０２はフリップフロップ、ｄ００はクラスタである。

【００４４】

このように、フリップフロップの同期／非同期指定だけでなく、ゲーテッドクロックのクラスタ結合も変更可能な回路記述ファイルを生成できる。しかも、低消費電力化のためのゲーテッドクロック化において、より大きなクラスタをクラスタ結合機能により簡単に実現することができるために、クラスタサイズの最適値を求める作業などに要する時間を大幅に短縮できる。

【００４５】

上述したように、この発明の実施の形態に係るプリプロセッサ、集積回路の設計システム及び集積回路の設計方法では、フリップフロップのハードウェア記述において、同期リセット付きフリップフロップであるか、非同期リセット付きフリップフロップであるかを特定しない形式で記述する。この際、上記フリップフロップに関する記述は、既存のハードウェア記述言語（第２のハードウェア記述言語）で記述されたファイル（第１の回路記述ファイル）の中に、第１のハードウェア記述言語で記述する。このフリップフロップに関する記述が埋め込まれた回路記述ファイルを変換プログラムＡ（ＰｒｏｇｒａｍＡ）で処理すると、フリップフロップに関する記述部は既存のハードウェア記述言語に変換されて出力される。このとき変換プログラムＡに与える引数により、同期リセット方式のフリップフロップ記述を出力させることができる。同様に、非同期リセット方式のフリップフロップ記述を出力させることもできる。

【００４６】

このように、ハードウェア記述言語の変換を変換プログラムＡにより自動化することにより、「同期リセット」と「非同期リセット」のそれぞれの方式で記述されたハードウェア記述を簡単に得ることができる。また、例えば「同期リセット」方式の記述側を検証すれば、「非同期リセット」方式の記述の方も検証されたことになるので、２つのハードウェア記述の検証にかかる時間を半分にすることができる、これにより、ＬＳＩの開発等において開発期間を短縮することができる。

【００４７】

また、フリップフロップのクロックをゲーテッドクロック化する指定を変換プログラムＡに対して指定することが可能であり、この場合にはフリップフロップのクロックをゲーティングした記述が生成される。更に、フリップフロップの記述Ｘおよび記述Ｙが存在する場合に、記述Ｘと記述Ｙのクロックを共通化し、共通化されたクロックに対してゲーテッドクロック化を行うこともできる（クラスタ結合）。

【００４８】

上述したように、様々なコンフィギュレーションに従ったＲＴＬ記述を容易に得ることが可能となる。

【００４９】

従って、上記のような構成並びに設計方法によれば、フリップフロップの同期／非同期指定及びゲーテッドクロックのクラスタ結合を変更可能な回路記述ファイルをプリプロセッサ２３で生成できるので、開発期間の短縮とコストの低減を図ることが可能となる。しかも、人手による作業ではバグの混入による修正が必要になり、修正のたびに機能検証作業が発生するが、自動化により検証作業が不要となり、この点からも開発期間の短縮やコストの低減が図れる。

【００５０】

なお、上記実施の形態では、説明を分かりやすくするために、回路記述ファイル２１とプリプロセッサ制御ファイル２２が別のファイルの場合を例に取って説明したが、１つのファイルにまとめても良いのは勿論である。

【００５１】

以上実施の形態を用いてこの発明の説明を行ったが、この発明は上記実施の形態に限定されるものではなく、実施段階ではその要旨を逸脱しない範囲で種々に変形することが可能である。また、上記実施の形態には種々の段階の発明が含まれており、開示される複数の構成要件の適宜な組み合わせにより種々の発明が抽出され得る。例えば実施の形態に示される全構成要件からいくつかの構成要件が削除されても、発明が解決しようとする課題の欄で述べた課題の少なくとも１つが解決でき、発明の効果の欄で述べられている効果の少なくとも１つが得られる場合には、この構成要件が削除された構成が発明として抽出され得る。

【００５２】

【発明の効果】

以上説明したように、この発明によれば、フリップフロップの同期／非同期指定及びゲーテッドクロックのクラスタ結合を変更可能な回路記述ファイルを生成できるプリプロセッサが得られる。

【００５３】

また、開発期間を短縮し、コストを低減できる集積回路の設計システムとこのシステムによる集積回路の設計方法が得られる。

【図面の簡単な説明】

【図１】　この発明の実施の形態に係るプリプロセッサ及び集積回路の設計システムについて説明するためのもので、システム全体の概略構成を示す機能ブロック図。

【図２】　図１に示したシステムにおけるプリプロセッサの概略的な動作を示すフローチャート。

【図３】　図１に示したシステムにおけるプリプロセッサの動作を示すもので、第１ステージの処理サブルーチンを示すフローチャート。

【図４】　図１に示したシステムにおけるプリプロセッサの動作を示すもので、拡張記述解析サブルーチンを示すフローチャート。

【図５】　図１に示したシステムにおけるプリプロセッサの動作を示すもので、第２ステージの処理サブルーチンを示すフローチャート。

【図６】　図１に示したシステムにおけるプリプロセッサの動作を示すもので、タグＢの処理サブルーチンを示すフローチャート。

【図７】　図１に示したシステムにおけるプリプロセッサの動作を示すもので、クラスタ結合サブルーチンを示すフローチャート。

【図８】　図２乃至図７に示したプリプロセッサによるフリップフロップのリセットの同期／非同期指定を変更する場合の入力と出力の関係について説明するための図。

【図９】　図２乃至図７に示したプリプロセッサによるクラスタ結合機能を使用しない場合の入力ファイルと出力ファイルとの関係について説明するための図。

【図１０】　図１に示した集積回路の設計システムを用い、図２乃至図７に示したプリプロセッサによるクラスタ結合機能を使用しないで形成した回路の構成例を示す回路図。

【図１１】　図２乃至図７に示したプリプロセッサによるクラスタ結合機能を使用した場合の入力ファイルと出力ファイルとの関係について説明するための図。

【図１２】　図１に示した回路システムを用い、図２乃至図７に示したプリプロセッサによるクラスタ結合機能を使用して形成した回路の構成例を示す回路図。

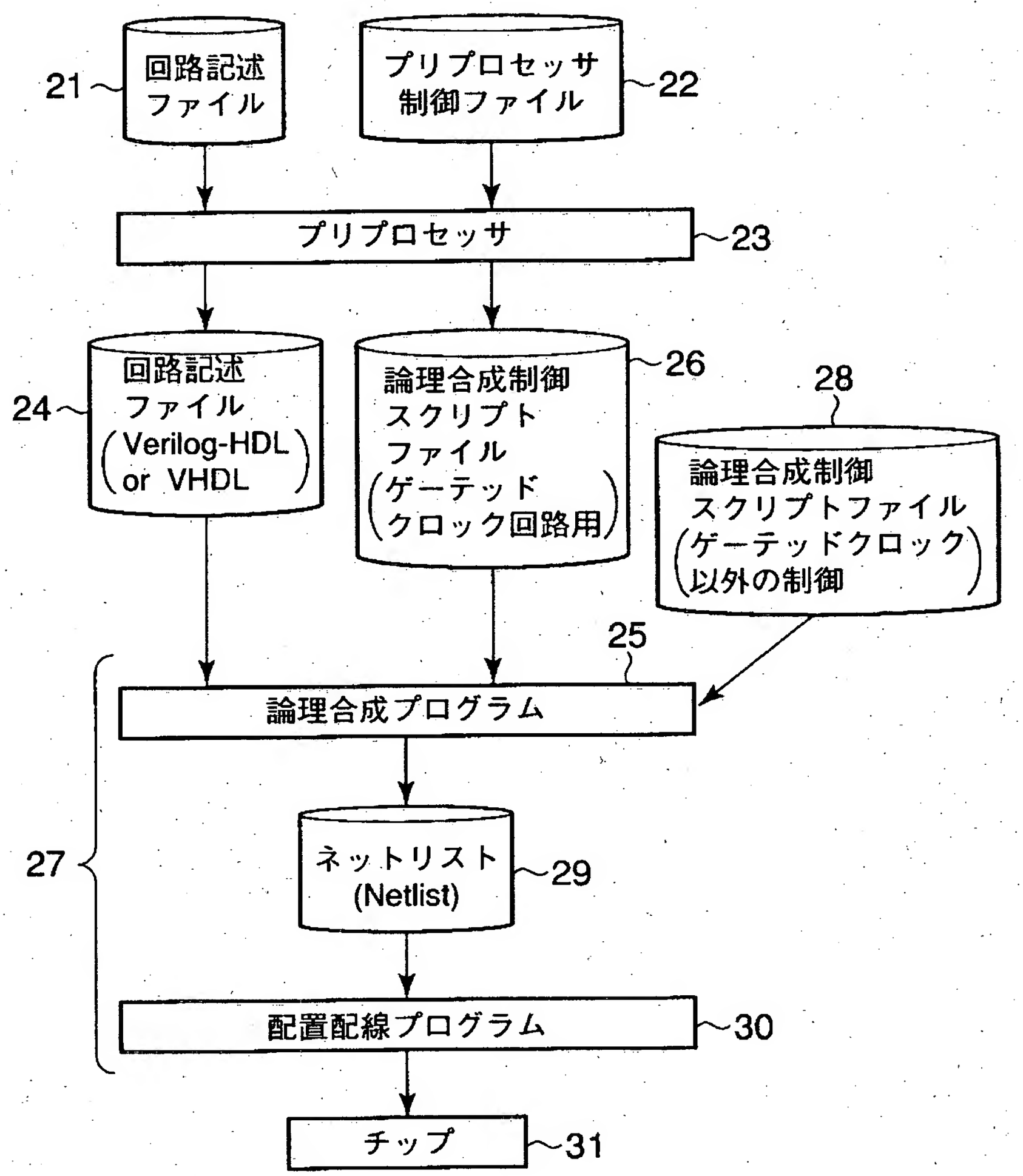
【図１３】　従来の集積回路の設計システム及び集積回路の設計方法について説明するための機能ブロック図。

【符号の説明】

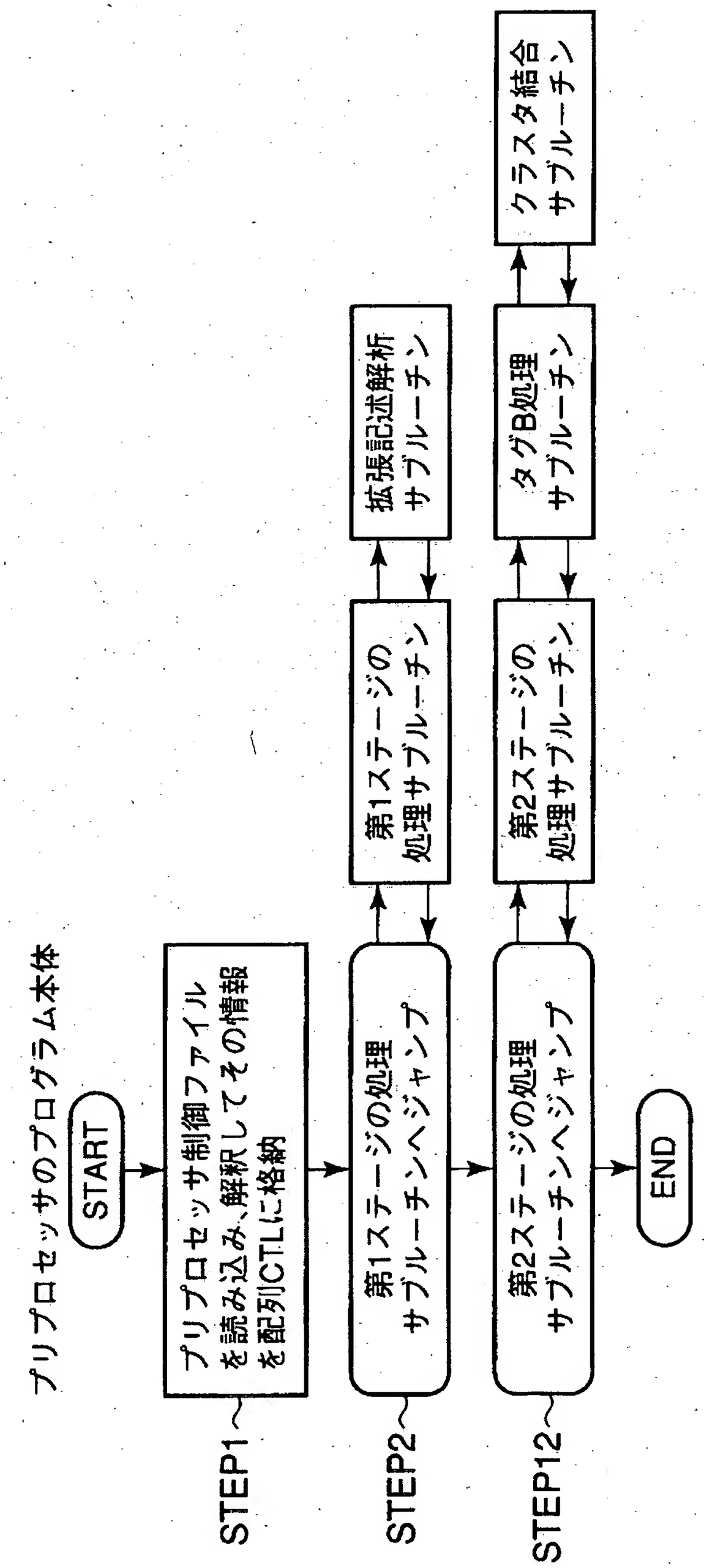
２１…回路記述ファイル（第１の回路記述ファイル）、２２…プリプロセッサ制御ファイル、２３…プリプロセッサ、２４…回路記述ファイル（第２の回路記述ファイル）、２５…論理合成プログラム、２６…論理合成制御スクリプトファイル、２７…論理合成ツール（コンピュータ）、２８…論理合成スクリプトファイル、２９…ネットリスト、３０…配置配線プログラム、３１…チップ。

【書類名】　　図面

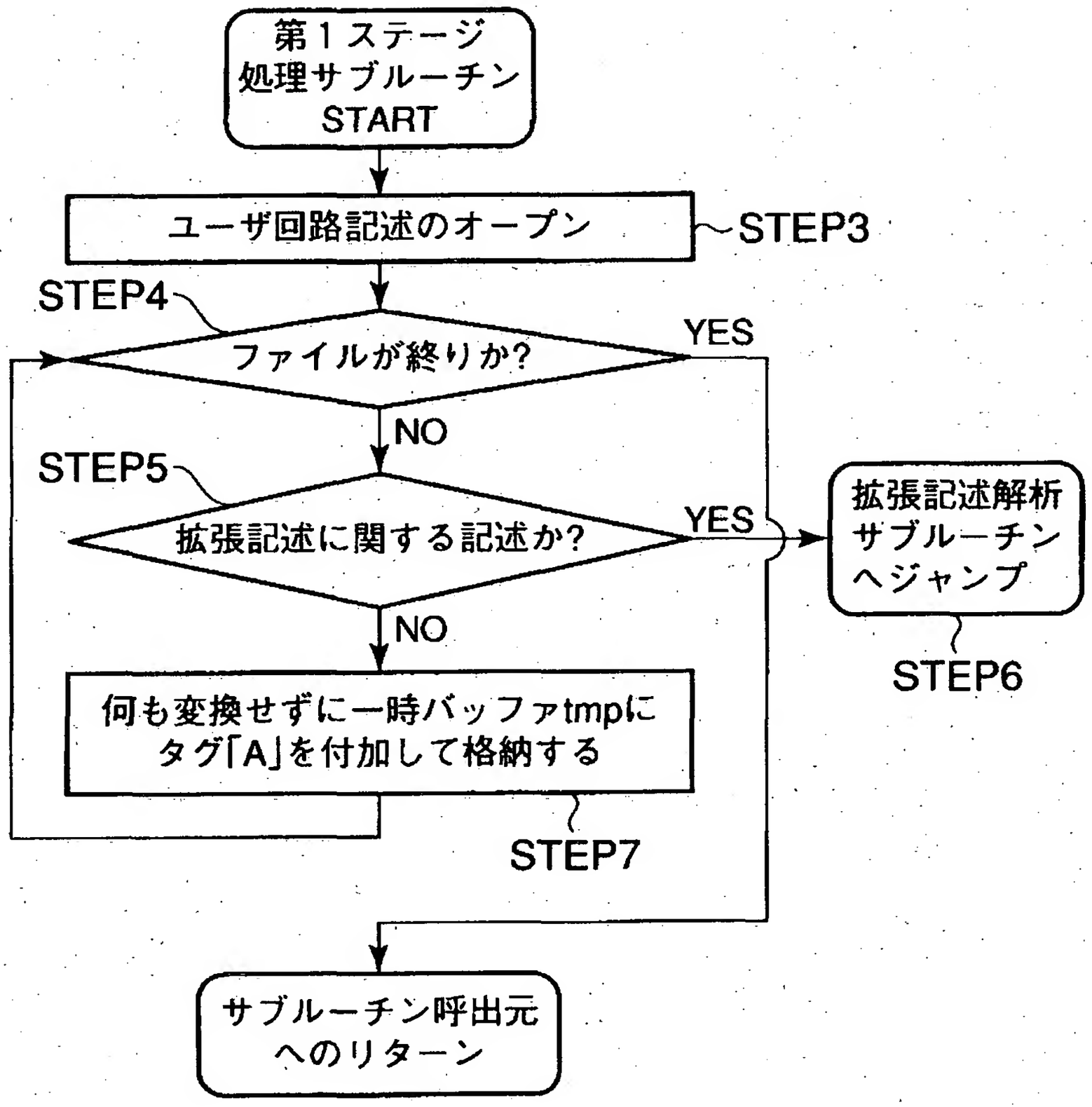
【図１】


回路記述
ファイル
21
プリプロセッサ
制御ファイル
22
プリプロセッサ
23
回路記述
ファイル
(Verilog-HDL
or VHDL)
24
論理合成制御
スクリプト
ファイル
(ゲーテッド
クロック回路用)
26
28
論理合成制御
スクリプトファイル
(ゲーテッドクロック
以外の制御
25
論理合成プログラム
27
ネットリスト
(Netlist)
29
配置配線プログラム
30
チップ
31

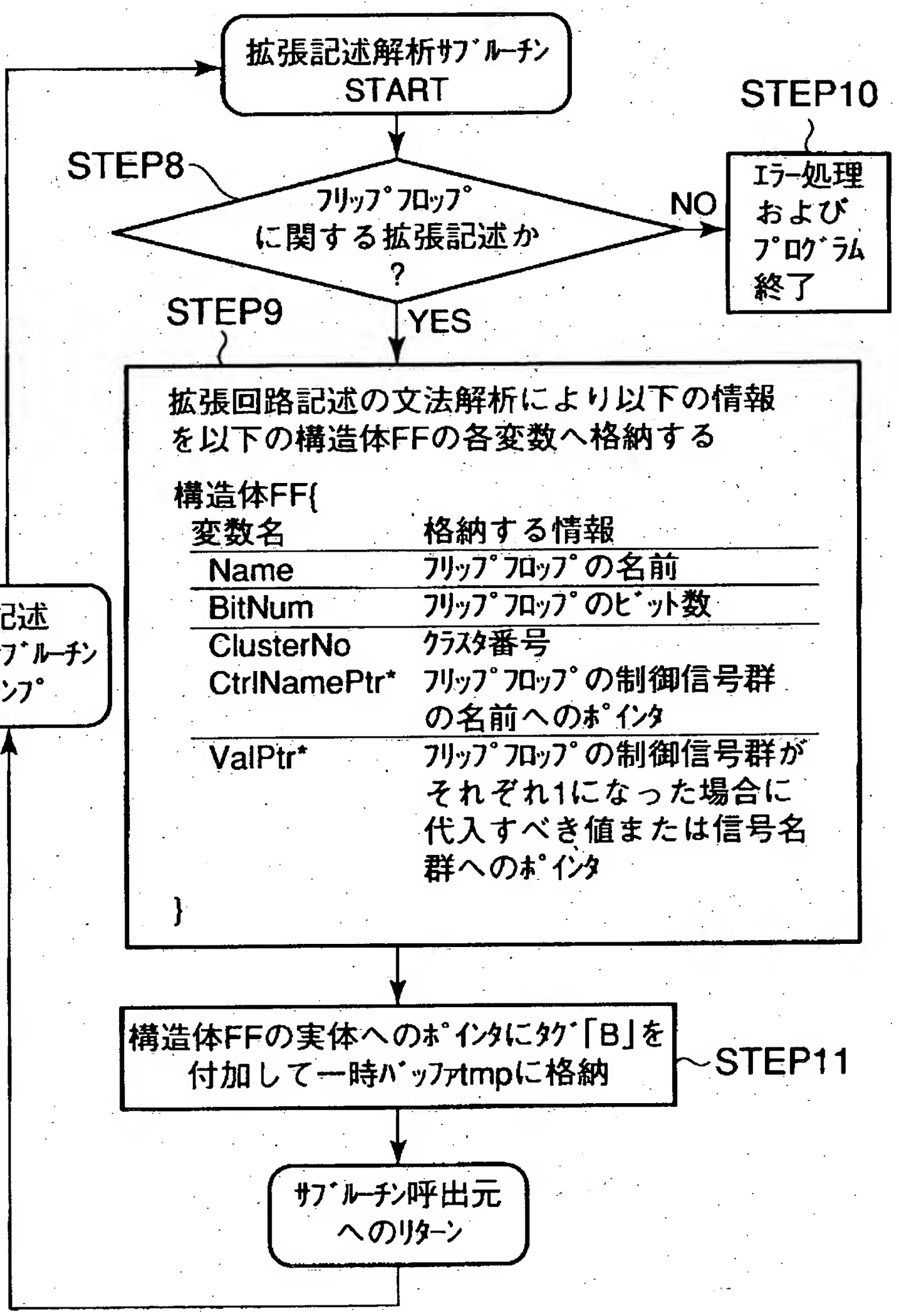
【図２】


プリプロセッサのプログラム本体
START
プリプロセッサ制御ファイル
を読み込み、解釈してその情報
を配列CTLに格納
STEP1
第1ステージの
処理サブルーチンへジャンプ
STEP2
第2ステージの
処理サブルーチンへジャンプ
STEP12
END
クラスタ結合
サブルーチン
拡張記述解析
サブルーチン
第1ステージの
処理サブルーチン
タグB処理
サブルーチン
第2ステージの
処理サブルーチン

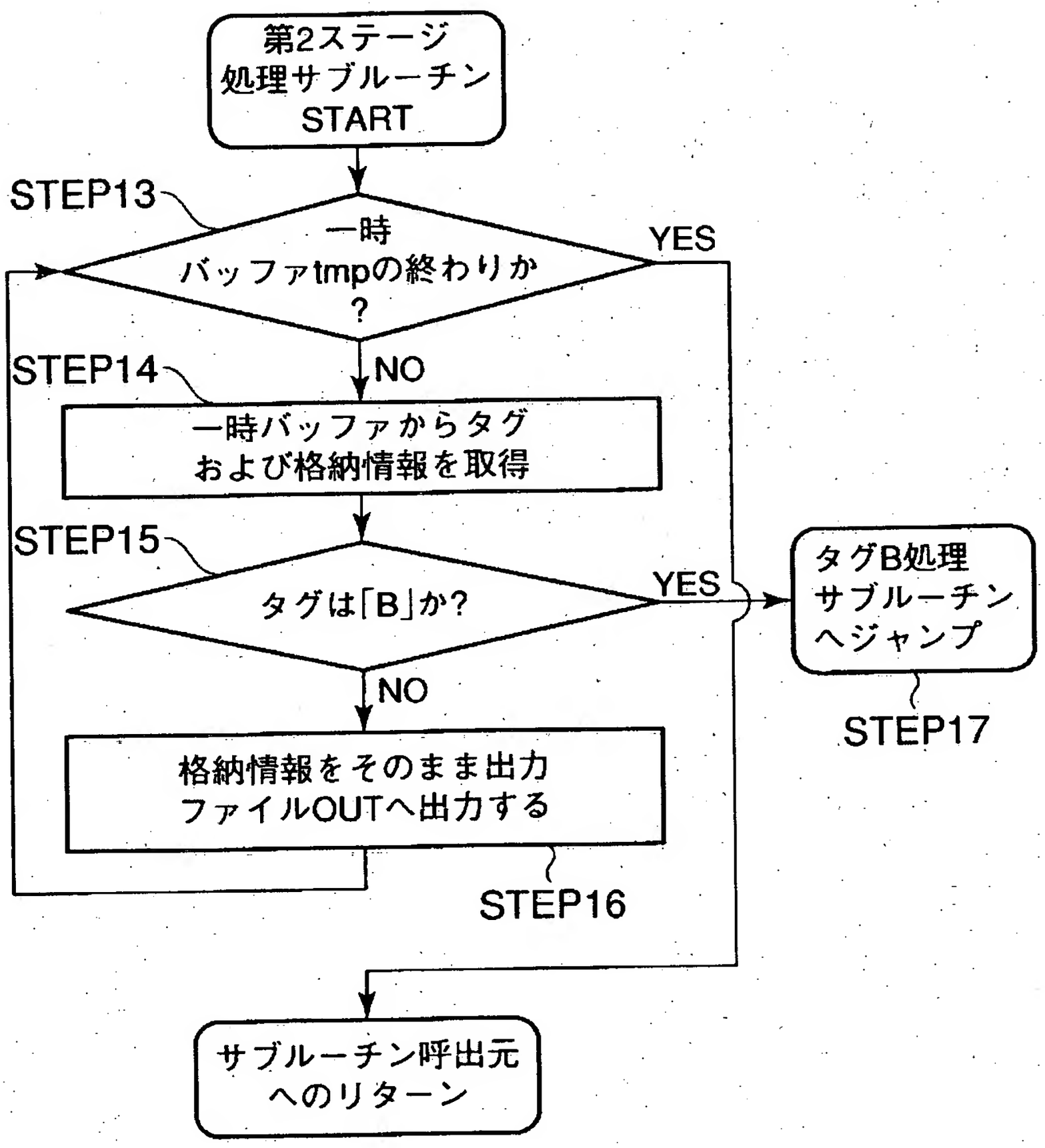
【図３】


第１ステージ
処理サブルーチン
START
ユーザ回路記述のオープン
STEP3
STEP4
ファイルが終りか?
YES
NO
STEP5
拡張記述に関する記述か?
YES
NO
拡張記述解析
サブルーチン
へジャンプ
STEP6
何も変換せずに一時バッファtmpに
タグ「A」を付加して格納する
STEP7
サブルーチン呼出元
へのリターン

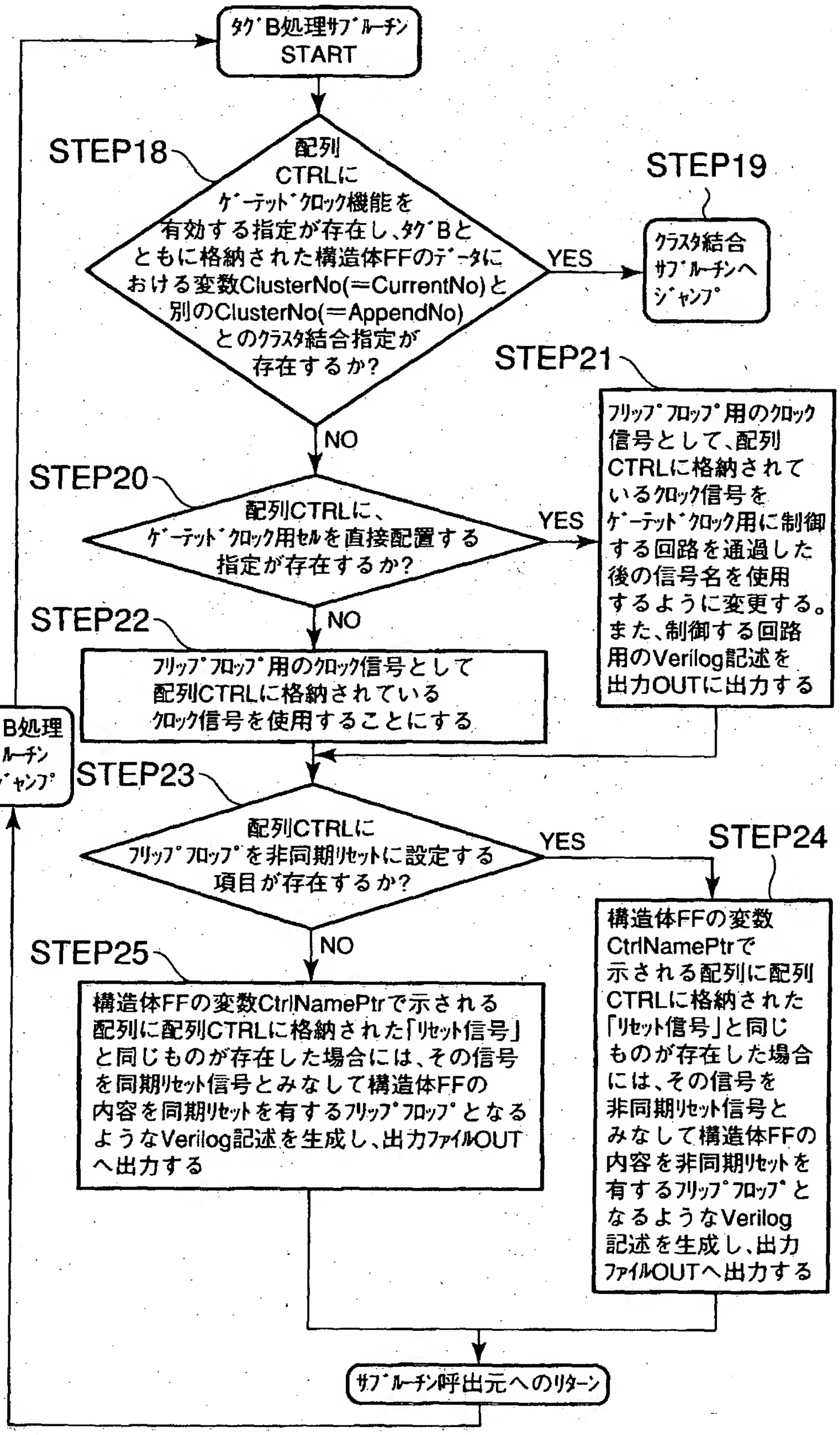
【図４】


拡張記述解析ｻﾌﾞﾙｰﾁﾝ
START
STEP8
ﾌﾘｯﾌﾟﾌﾛｯﾌﾟ
に関する拡張記述か
？
NO
STEP10
ｴﾗｰ処理
および
ﾌﾟﾛｸﾞﾗﾑ
終了
YES
STEP9
拡張回路記述の文法解析により以下の情報
を以下の構造体FFの各変数へ格納する
構造体FF{
変数名
格納する情報
Name
ﾌﾘｯﾌﾟﾌﾛｯﾌﾟの名前
BitNum
ﾌﾘｯﾌﾟﾌﾛｯﾌﾟのﾋﾞｯﾄ数
ClusterNo
ｸﾗｽﾀ番号
CtrlNamePtr*
ﾌﾘｯﾌﾟﾌﾛｯﾌﾟの制御信号群
の名前へのﾎﾟｲﾝﾀ
ValPtr*
ﾌﾘｯﾌﾟﾌﾛｯﾌﾟの制御信号群が
それぞれ1になった場合に
代入すべき値または信号名
群へのﾎﾟｲﾝﾀ
}
拡張記述
解析ｻﾌﾞﾙｰﾁﾝ
ﾍｼﾞｬﾝﾌﾟ
構造体FFの実体へのﾎﾟｲﾝﾀにﾀｸﾞ「B」を
付加して一時ﾊﾞｯﾌｧtmpに格納
STEP11
ｻﾌﾞﾙｰﾁﾝ呼出元
へのﾘﾀｰﾝ

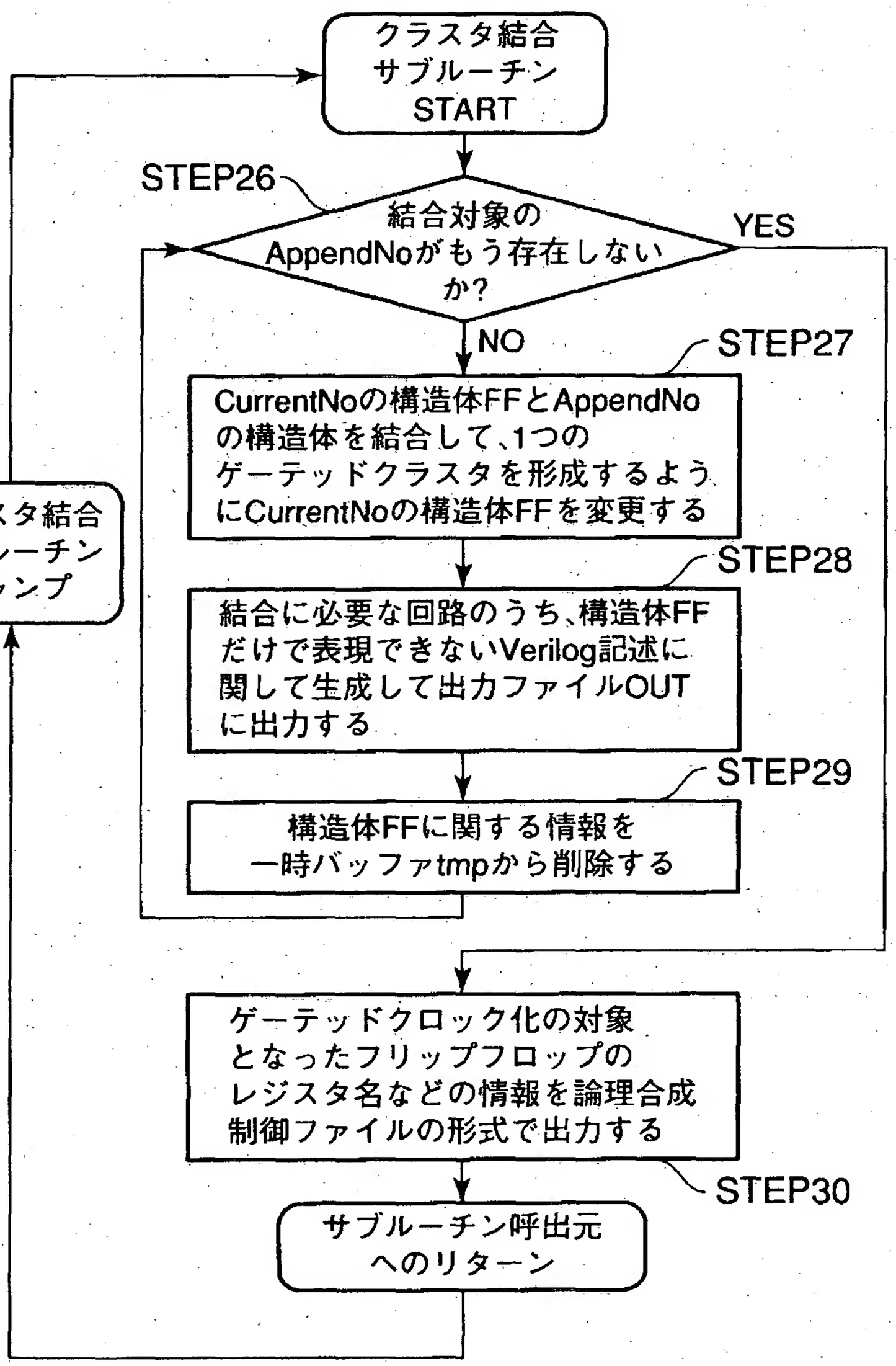
【図５】


第2ステージ
処理サブルーチン
START
STEP13
一時
バッファtmpの終わりか
？
YES
NO
STEP14
一時バッファからタグ
および格納情報を取得
STEP15
タグは「B」か?
YES
NO
タグB処理
サブルーチン
へジャンプ
STEP17
格納情報をそのまま出力
ファイルOUTへ出力する
STEP16
サブルーチン呼出元
へのリターン

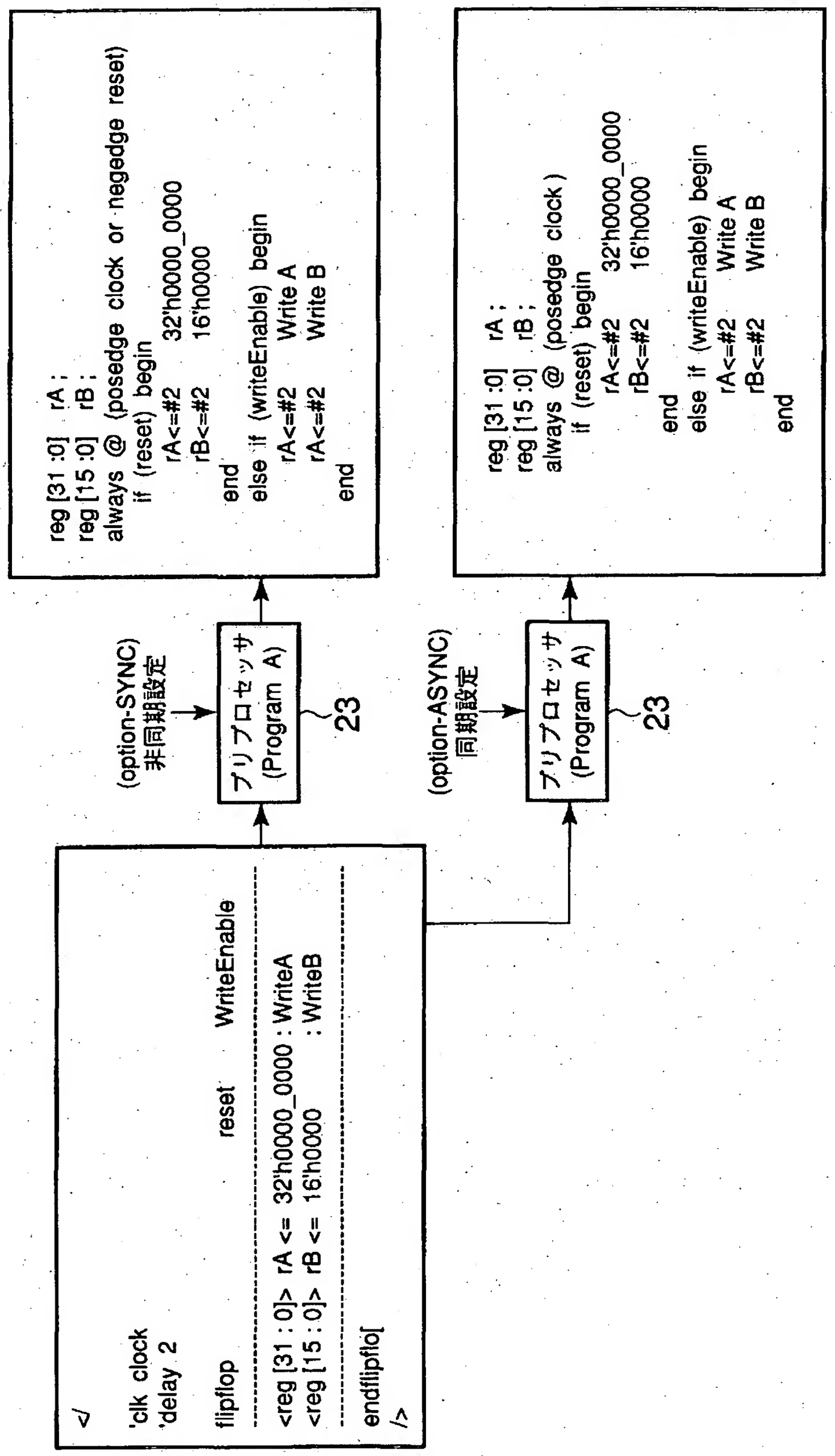
【図６】


ﾀｸﾞB処理ｻﾌﾞﾙｰﾁﾝ
START
STEP18
配列
CTRLに
ｹﾞｰﾃｯﾄﾞｸﾛｯｸ機能を
有効する指定が存在し、ﾀｸﾞBと
ともに格納された構造体FFのﾃﾞｰﾀに
おける変数ClusterNo(＝CurrentNo)と
別のClusterNo(＝AppendNo)
とのｸﾗｽﾀ結合指定が
存在するか?
YES
STEP19
ｸﾗｽﾀ結合
ｻﾌﾞﾙｰﾁﾝへ
ｼﾞｬﾝﾌﾟ
NO
STEP20
配列CTRLに、
ｹﾞｰﾃｯﾄﾞｸﾛｯｸ用ｾﾙを直接配置する
指定が存在するか?
YES
STEP21
ﾌﾘｯﾌﾟﾌﾛｯﾌﾟ用のｸﾛｯｸ
信号として、配列
CTRLに格納されて
いるｸﾛｯｸ信号を
ｹﾞｰﾃｯﾄﾞｸﾛｯｸ用に制御
する回路を通過した
後の信号名を使用
するように変更する。
また、制御する回路
用のVerilog記述を
出力OUTに出力する
NO
STEP22
ﾌﾘｯﾌﾟﾌﾛｯﾌﾟ用のｸﾛｯｸ信号として
配列CTRLに格納されている
ｸﾛｯｸ信号を使用することにする
ﾀｸﾞB処理
ｻﾌﾞﾙｰﾁﾝ
へｼﾞｬﾝﾌﾟ
STEP23
配列CTRLに
ﾌﾘｯﾌﾟﾌﾛｯﾌﾟを非同期ﾘｾｯﾄに設定する
項目が存在するか?
YES
STEP24
構造体FFの変数
CtrlNamePtrで
示される配列に配列
CTRLに格納された
「ﾘｾｯﾄ信号」と同じ
ものが存在した場合
には、その信号を
非同期ﾘｾｯﾄ信号と
みなして構造体FFの
内容を非同期ﾘｾｯﾄを
有するﾌﾘｯﾌﾟﾌﾛｯﾌﾟと
なるようなVerilog
記述を生成し、出力
ﾌｧｲﾙOUTへ出力する
NO
STEP25
構造体FFの変数CtrlNamePtrで示される
配列に配列CTRLに格納された「ﾘｾｯﾄ信号」
と同じものが存在した場合には、その信号
を同期ﾘｾｯﾄ信号とみなして構造体FFの
内容を同期ﾘｾｯﾄを有するﾌﾘｯﾌﾟﾌﾛｯﾌﾟとなる
ようなVerilog記述を生成し、出力ﾌｧｲﾙOUT
へ出力する
ｻﾌﾞﾙｰﾁﾝ呼出元へのﾘﾀｰﾝ

【図７】


クラスタ結合
サブルーチン
START
STEP26
結合対象の
AppendNoがもう存在しない
か?
YES
NO
STEP27
CurrentNoの構造体FFとAppendNo
の構造体を結合して、1つの
ゲーテッドクラスタを形成するよう
にCurrentNoの構造体FFを変更する
STEP28
結合に必要な回路のうち、構造体FF
だけで表現できないVerilog記述に
関して生成して出力ファイルOUT
に出力する
STEP29
構造体FFに関する情報を
一時バッファtmpから削除する
クラスタ結合
サブルーチン
へジャンプ
ゲーテッドクロック化の対象
となったフリップフロップの
レジスタ名などの情報を論理合成
制御ファイルの形式で出力する
STEP30
サブルーチン呼出元
へのリターン

【図８】


reg [31 :0] rA ;
reg [15 :0] rB ;
always @ (posedge clock or negedge reset)
if (reset) begin
rA<=#2 32'h0000_0000
rB<=#2 16'h0000
end
else if (writeEnable) begin
rA<=#2 Write A
rB<=#2 Write B
end
(option-SYNC)
非同期設定
プリプロセッサ
(Program A)
23
reg [31 :0] rA ;
reg [15 :0] rB ;
always @ (posedge clock)
if (reset) begin
rA<=#2 32'h0000_0000
rB<=#2 16'h0000
end
else if (writeEnable) begin
rA<=#2 Write A
rB<=#2 Write B
end
(option-ASYNC)
同期設定
プリプロセッサ
(Program A)
23
WriteEnable
: WriteA
: WriteB
reset
<reg [31 : 0]> rA <= 32'h0000_0000 :
<reg [15 : 0]> rB <= 16'h0000
'clk clock
'delay 2
flipflop
endflipflo[

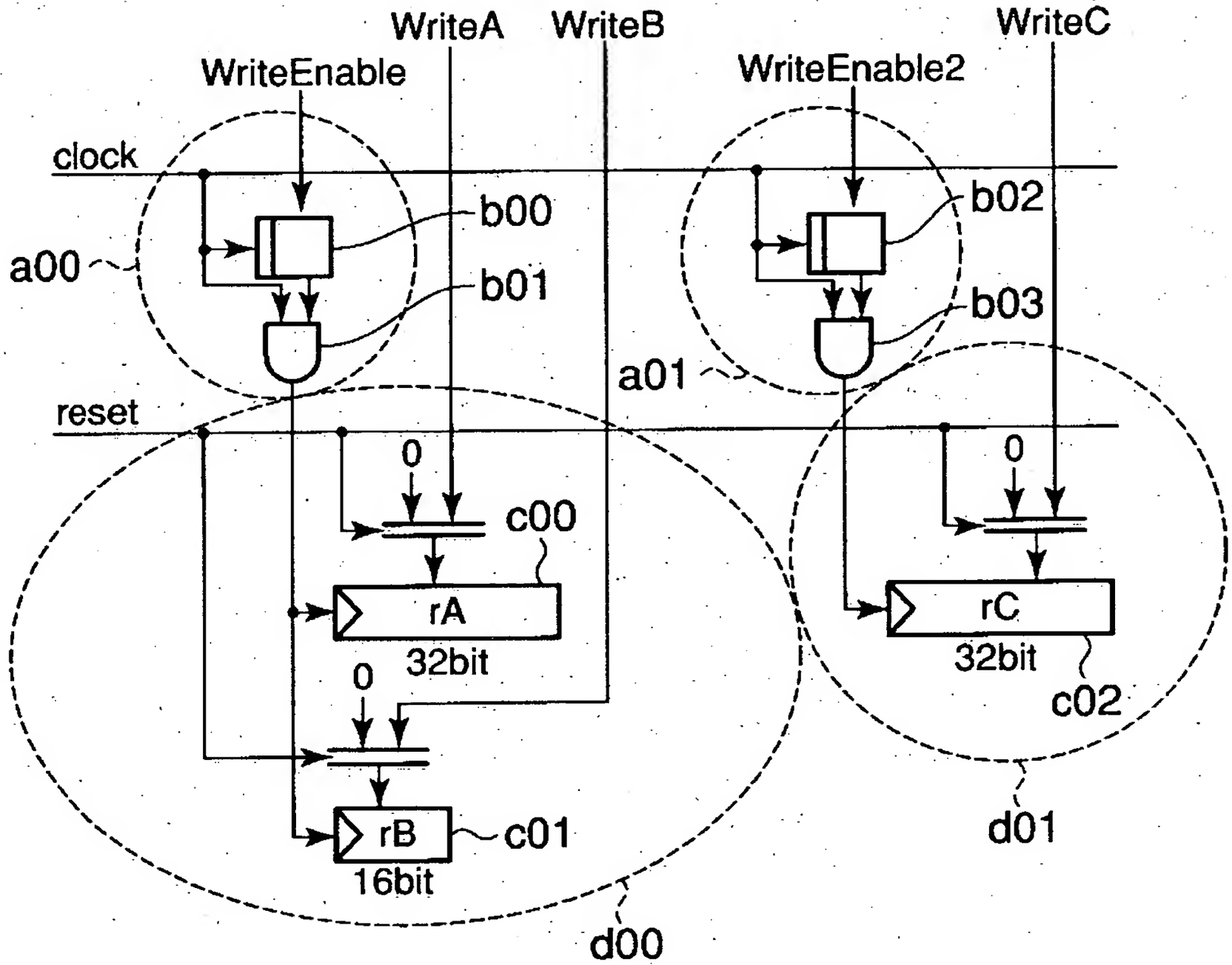
【図９】

```
</

'clk clock
'delay 2

flipflop                          reset        WriteEnable
-------------------------------------------------------------------
<reg [31 : 0]> rA <= 32'h0000_0000 : WriteA
<reg [15 : 0]> rB <= 16'h0000      : WriteB
-------------------------------------------------------------------
endflipflop<name=FF00>
                  * * *
flipflop                          reset        WriteEnable2
-------------------------------------------------------------------
<reg [31 : 0]> rC <= 32'h0000_0000 : WriteC
-------------------------------------------------------------------
endflipflop<name=FF01>
                  * * *
/>
```

23～プリプロセッサ(Program A)←enable gated-clock

```
wire wlatched00 ;
wire gated_clock00 ;

wire wlatched01 ;
wire gated_clock01;

GC_LATCH gc_latch(wlatched00, clock, WriteEnable) ;
GC_GATE   gc_gate(gated_clock00, clock, wlatched00);
reg [31 : 0]> rA ;
reg [15 : 0]> rB ;
always @ (posedge gated_clock00)
    if (reset) begin
        rA<=#2      32'h0000_0000 ;
        rB<=#2      16'h0000 ;
    end
    else begin
        rA<=#2      Write Agc ;
        rA<=#2      Write Bgc ;
    end

GC_LATCH gc_latch(wlatched01, clock, WriteEnable2) ;
GC_GATE   gc_gate(gated_clock001, clock, wlatched01);

reg [31 :0]   rA ;
reg [15 :0]   rB ;
always @ (posedge gated_clock01 )
    if (reset) begin
        rC<=#2      32'h0000_0000 ;
    end
    else begin
        rC<=#2      WriteCgc ;
    end
```

【図１０】


WriteA
WriteB
WriteC
WriteEnable
WriteEnable2
clock
b00
b01
b02
b03
a00
a01
reset
0
c00
rA
32bit
0
rB
c01
16bit
rC
32bit
c02
d00
d01

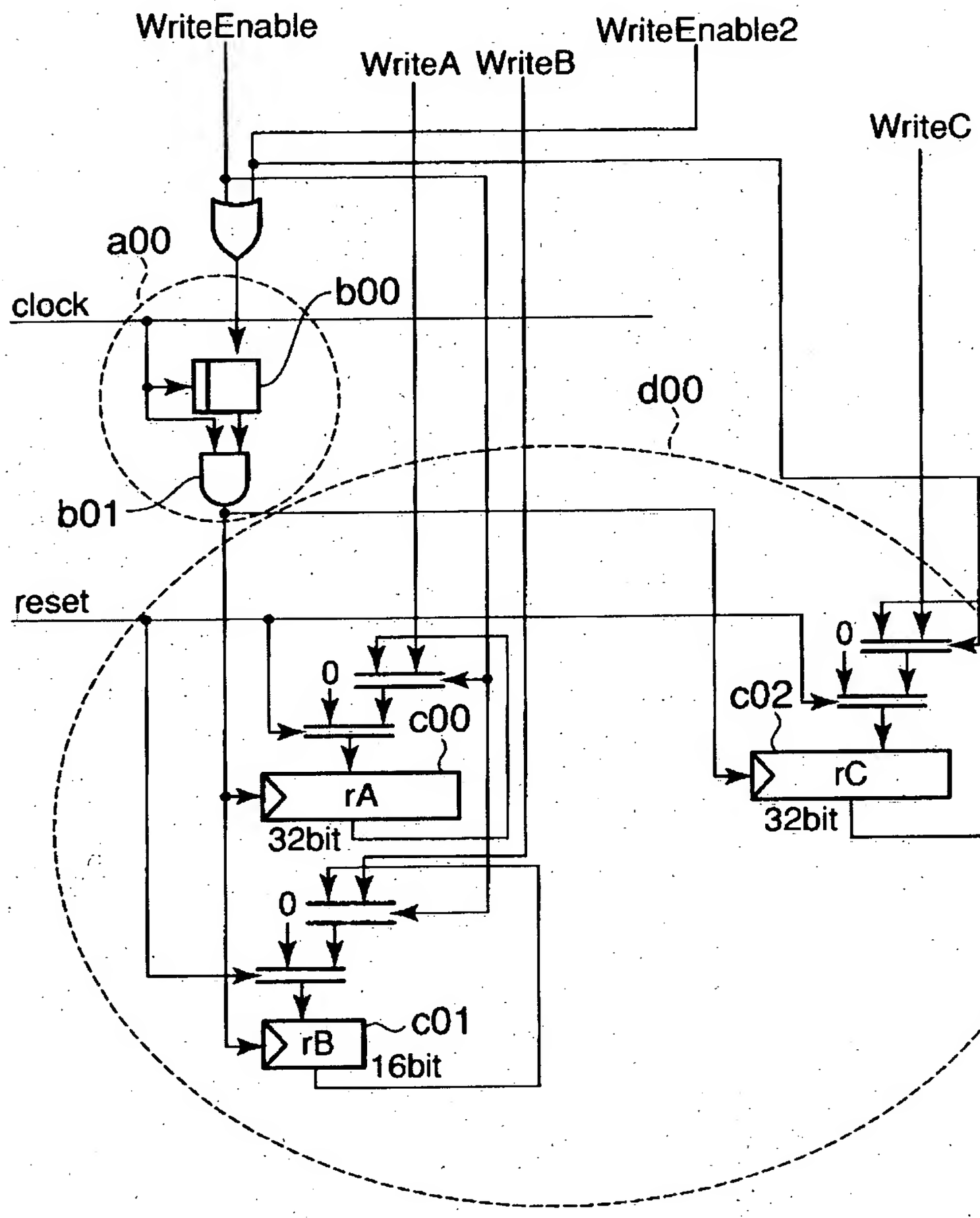
【図１１】

```
</

'clk clock
'delay 2

flipflop                          reset        WriteEnable
-------------------------------------------------------------------
<reg [31 : 0]> rA <= 32'h0000_0000 : WriteA
<reg [15 : 0]> rB <= 16'h0000      : WriteB
-------------------------------------------------------------------
endflipflop<name=FF00>
           ***
flipflop                          reset        WriteEnable2
-------------------------------------------------------------------
<reg [31 : 0]> rC <= 32'h0000_0000 : WriteC
-------------------------------------------------------------------
endflipflop<name=FF01>
           * * *
/>
```

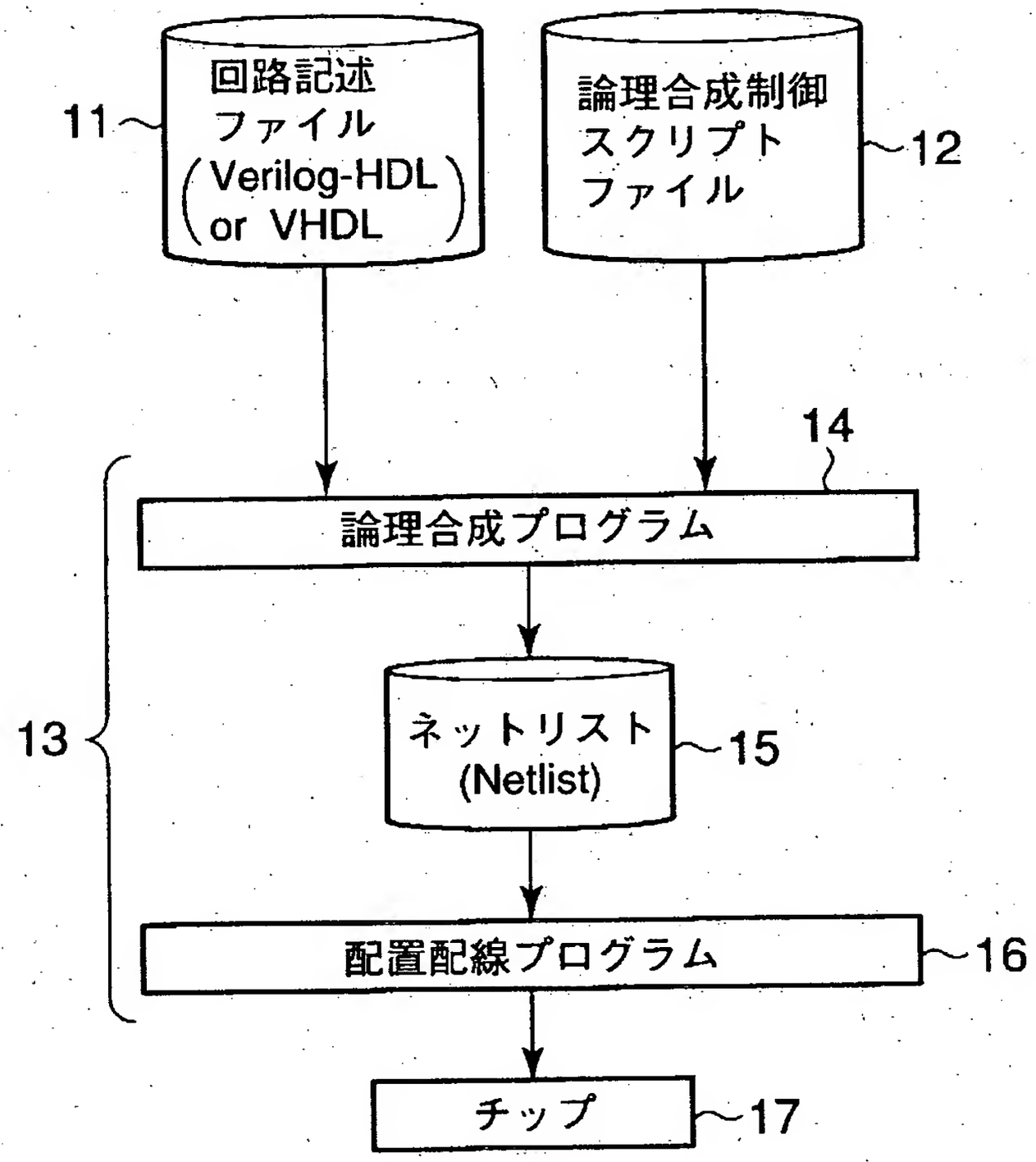
23～ プリプロセッサ (Program A) ← enable gated-clock
← grouping {FF00, FF01)

```
wire  [31 : 0]  WriteAgc ;
wire  [15 : 0]  WriteBgc ;
wire  [31 : 0]  WriteCgc ;
wire  gc_enable00 ;
wire  wlatched00 ;
wire  gated_clock00 ;

assign  WriteAgc = WriteEnable   ?  WriteA : rA ;
assign  WriteBgc = WriteEnable   ?  WriteB : rB ;
assign  WriteCgc = WriteEnable2  ?  WriteC : rC ;

assign  gc_enable00  = WriteEnable  |  WriteEnable2 ;

GC_LATCH  gc_latch(wlatched00, clock, gc_enble00) ;
GC_GATE   gc_gate (gated_clock00, clock, wlatched00) ;

reg  [31 : 0]  rA ;
reg  [15 : 0]  rB ;
always  @  (posedge  gated_clock00)
    if  (reset)  begin
        rA  <=  #2    32'h0000_0000 ;
        rB  <=  #2    16'h0000 ;
        rC  <=  #2    32'h0000_0000 ;
    end
    else  if (WriteEnable)  begin
        rA  <=  #2  WriteAgc ;
        rB  <=  #2  WriteBgc ;
        rC  <=  #2  WriteCgc ;
    end
```

【図１２】


WriteEnable
WriteA
WriteB
WriteEnable2
WriteC
a00
clock
b00
b01
d00
reset
0
c00
rA
32bit
0
c02
rC
32bit
0
rB
c01
16bit

【図１３】


回路記述
ファイル
(Verilog-HDL
or VHDL)
11
論理合成制御
スクリプト
ファイル
12
14
論理合成プログラム
13
ネットリスト
(Netlist)
15
配置配線プログラム
16
チップ
17

【書類名】　　要約書

【要約】

【課題】　フリップフロップの同期／非同期指定及びゲーテッドクロックのクラスタ結合を変更可能な回路記述ファイルを生成できるプリプロセッサを提供することを目的としている。

【解決手段】　プリプロセッサ２３は、ユーザによって記述されたプリプロセッサ制御ファイル２２に基づいて、ユーザによって記述され、第１のハードウェア記述言語と第２のハードウェア記述言語が混在する第１の回路記述ファイル２１の処理を行い、前記第１の回路記述ファイル２１における第１のハードウェア記述言語で記述された少なくとも一部を第２のハードウェア記述言語に変換して第２の回路記述ファイル２４を生成し、出力することを特徴としている。これによって、人手を介在することなく、フリップフロップの同期／非同期指定及びゲーテッドクロックのクラスタ結合を変更可能な回路記述ファイルを生成できる。

【選択図】　　図１

出　願　人　履　歴　情　報

識別番号　　〔０００００３０７８〕

１．変更年月日　　２００１年　７月　２日
　［変更理由］　　住所変更
　　住　所　　東京都港区芝浦一丁目１番１号
　　氏　名　　株式会社東芝